CHAPITRE III : PREMIERS PAS EN PYTHON

PRESENTATION

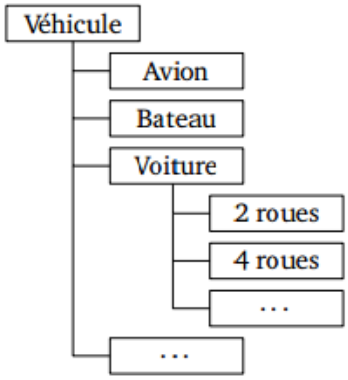
Le langage python est un langage, orienté objet, interprété, de haut niveau,

développé en 1989 par Guido van Rossum.

En 2005, il a été engagé par Google pour ne travailler que sur python.

Le langage python est . . .

- **entièrement gratuit**
- **portable** : Un même programme s'exécute sur Linux, Windows, Mac Os . . .
- **interprété** : Pas de phase de compilation qui traduit le programme en langage machine.
- **orienté objet** : sans l'imposer.



- **de haut niveau :**

– La syntaxe permet de se libérer des détails inhérents au fonctionnement de l'ordinateur.

– Python possède un garbage collector : destruction automatique des objets qui ne sont plus utilisés.

– et des structures de données complexes (dictionnaires, ..) éloignées des types numériques standards.

- **modulaire :**

Autour d'une définition concise du langage, de nombreuses librairies ou modules ont été développées.

- **à syntaxe positionnelle :** L'indentation fait partie du langage.

- **Points forts - points faibles**

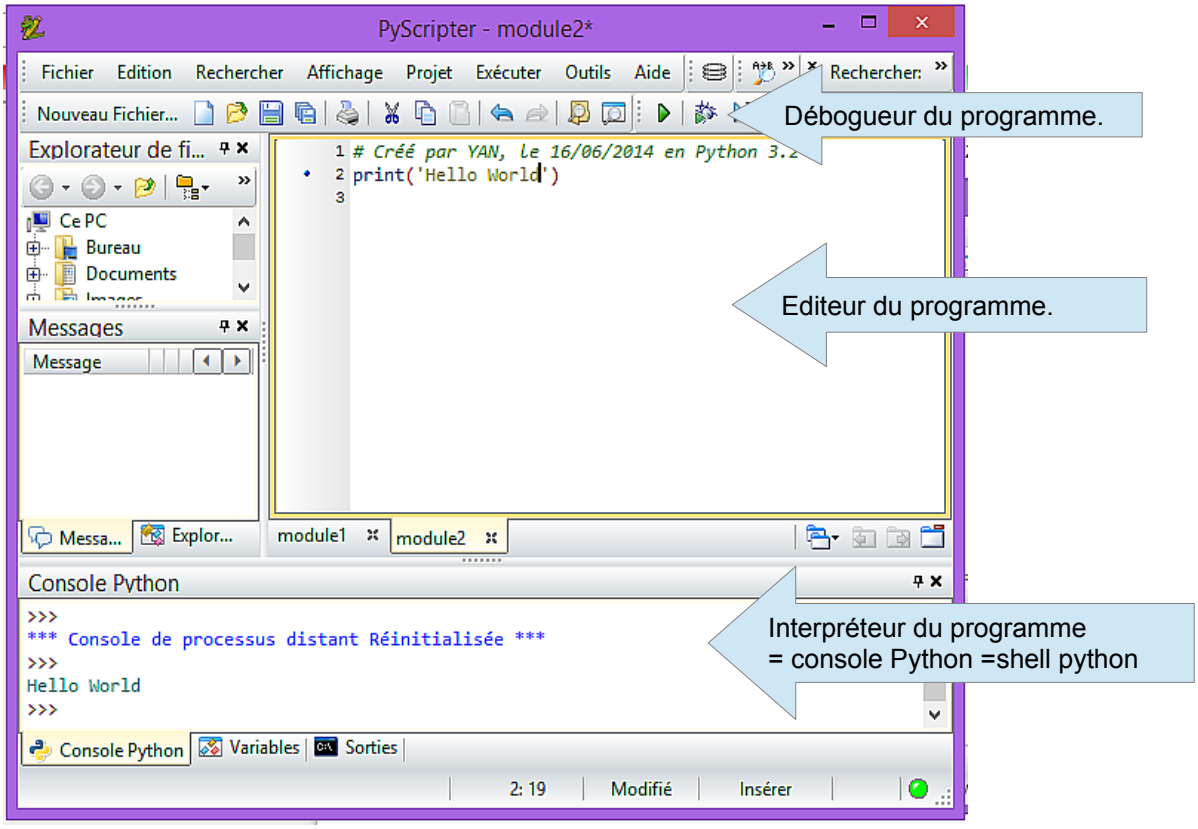
Inconvénients	Avantages
• vitesse d'exécution plus lente que le langage C++ • moins utilisé que le C++ ou le Java	• syntaxe plus simple que celle de Java ou du C++ – python est un langage plus simple à apprendre – amélioration significative des temps de développement • pas de déclaration de types, de variables, ... • le code python est trois à cinq fois plus court que le code Java équivalent • le code python est de cinq à dix fois plus court que le code C++ correspondant

INSTALLATION : sous windows

Nous utiliserons la distribution Edupython, basée sur Python 3.2. Elle inclue un IDE (Integrated Development Environment) nommé Pyscripteur (contenant un éditeur, un interpréteur, un débogueur) et d'utiles bibliothèques de fonctions. Enfin elle est portable et téléchargeable ici .

NB : De nombreux programmes existent en version Python 2.x, mais la compatibilité n'est pas assurée.

Exécuter EduPython. L'interface Pyscripter est composée de plusieurs zones :



Ex 1 : Premières opérations

L'avantage d'un langage interprété est que l'on peut directement tester une commande avec l'interpréteur. C'est très utile en débogage pour tester des instructions ! Cela fonctionne un peu comme une calculatrice : l'invite de commande >>> indique que vous avez la main. Tester les commandes suivantes **dans la Console Python (** ou interpréteur**) en bas** . Les commentaires après # sont ignorés.

```
>>>15+3
>>>2 – 17                # Les espaces sont ignorés, sauf au tout début
>>>8+3*4                 # Les priorités sont respectées
>>>(8+3)*4
>>>20/3
>>>20//3                 # Quelle est cette opération spéciale ?
>>>20%3                  # Quelle est cette opération spéciale ?
>>>3**2                  # Quelle est cette opération ?
>>>3**3                  # Note : en tapant sur la flèche du haut, on remonte dans les instructions.
```

Comme vous le constatez les quatre opérations de base (et priorités) sont fonctionnelles.

NOTION DE VARIABLE

Il n'y a pas que des nombres (entiers, décimaux) en programmation. Voici d'autres types de variables.

- **INT** et **FLOAT** représentent les **entiers** et les **décimaux**
- **STRING** représente les caractères et les **chaînes de caractères**
- **BOOLEAN** représente les booléens, qui ne prennent que la valeur **TRUE** ou **FALSE**.
- **LIST** représente les séries **d'éléments divers** : entiers , caractères, ou booléens, …
- a **=** *truc* signifie affecte la valeur *truc* à la variable a.

Avec Python il n'est pas nécessaire de déclarer le type des variables : elles sont dynamiquement typées à l'instant de leur première affectation. L'instruction type() renvoie le type (ou classe) de toute variable.

Ex 2 : Deviner le type des variables. (Toujours dans la console Python !).

Tapez les instructions comme suit, compléter le commentaire , puis vérifier avec enter

>>>a=3 >>>type(a) # type …. ? ← votre proposition, puis enter …. ?
>>>type(a>4) # …. ← votre proposition, puis enter >>>type(a==3) # …. ← votre proposition, puis enter
>>>b=5.5 >>>type(b) #….
>>>c=' 47 ' >>>type(c) # ….
>>>d= ' Douze ' >>>type(d) #….
>>>e=[1,2,a,b, True, 'Salut'] >>>type(e) #….

Ex 3 : **Opérations sur les variables.** Conserver a=3, b=5.5 et c='47' et procéder comme en A2

>>>a+b	# .....<--- votre proposition	Additionne deux nombres
>>>c+d	# .....<--- votre proposition	**Concatène** les chaînes c et d
>>>print(c+d)	….	Imprime ce qui est **entre guillemets**
>>>a+c	….	Ces deux types ne peuvent pas s'ajouter
>>> a*2	….	Multiplier par deux
>>>a**2	….	Puissance 2
>>>a**3	….	Puissance 3
>>>c*2	….	Reproduit la chaîne deux fois
>>>c*3	….	Reproduit la chaîne trois fois
>>> not a==4	….	C'est logique.
>>> a==3 and b==5	….	Les 2 conditions sont elles réalisées ?
>>> a==3 or b==5	….	Une des 2 conditions est-elle réalisée ?

Noter le signe égal différent pour une affectation « = » ou pour une condition « == »

Ex 4 : Le transtypage est une opération souvent nécessaire qui consiste à traduire le type d'une variable, en un autre type : par exemple float() traduit en décimal, un type chaîne ou un type entier. Conserver a=3, b=5.5 et c='47' . Observer, deviner et tester comme précédemment.

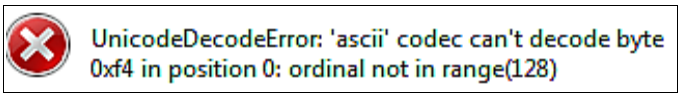
```
>>> float(a)              # ……..                ici float() traduit un entier en un décimal
>>> print( float (a))     #  ……..
>>> str(a)                #……..                ici str() traduit un entier en une chaîne
>>> print(str(a)+str(b))   ……..…….
>>> int(c)                 ……..                 ici int() traduit une chaîne en un entier
>>> a+int(c)               ……..
```

NOMS DE VARIABLES, DE FICHIERS

Les noms de variables doivent commencer par une lettre, mais ensuite on peut utiliser à peu près ce que l'on veut sauf les caractères spéciaux et les noms réservés aux instructions de Python.

Conseil pour vos futurs programmes employer des noms de variables évocateurs (Ex : ma_variable) . Ce sera plus clair même si c'est un peu plus long.

Les nom de fichiers doivent être sans accents, un message d'erreur de ce type peut survenir.



Si un accent gène dans le chemin du fichier il faudra l'enlever, ou sinon changer le fichier de place. Python 3.x pose moins de soucis car l'encodage est par défaut Utf-8 (Édition et Format de fichier) sinon taper `# -*- coding: utf-8 -*-` en début de code

PREMIERS PROGRAMMES

Placer le curseur dans la **zone module** (éditeur du programme) et taper des opérations semblables aux précédentes en passant à la ligne(par exemple a= 3, puis en dessous type(a)) puis cliquer sur la flèche verte (exécution du programme, ou Ctrlf 9)

Normalement il ne se passe pas grand chose ! Pour afficher quelque-chose il faut le demander à Python

La fonction **print()** sert à afficher et s'utilise ainsi : **print('*la valeur a est*', a)**

On affiche différents objets et commentaires (entre guillemets) en les séparant par des virgules

Pour écrire successivement sur la même ligne faire : print(a, **end=' '**)

Ex 5 : Tradition

Écrire un programme pour afficher "hello world". L'exécuter (Exécuter ou Ctrl F9 ou )

Compléter le en affectant une valeur à la variable a et faire afficher 'le carré de a vaut ...'

L'enregistrer sous "A5.py".

A6 : **Primaire**

```
Le nombre a vaut 2034
Nombre des centaines 20
Nombre des dizaines 3
Nombre des unités 4
```

Écrire un programme affichant le nombre de centaines, de dizaines et d'unités d'un entier a quelconque, comme ci contre.

L'enregistrer sous A6.py . Faites des tests avec plusieurs nombres. Penser à utiliser la division entière //

Comment entrer des données dans un programme ?

La fonction **input()** s'utilise ainsi **a=input("***entrez un nombre***")**

avec 'a' la mémoire où sera stockée la donnée et entre guillemets le message à afficher

Attention, **input() retourne toujours une chaîne de caractères**, a est donc ici de type string.

Remarque : pour entrer un type int ou float, il faudra donc procéder à un transtypage.

Pour cela on peut composer les fonctions de Python. Exemple : a= int (input("entrez un nombre")).

Ex 7 : **Au choix**

Reprendre l'exercice A6 mais c'est l'utilisateur qui entre un nombre entier.

On peut assigner en une seule ligne diverses valeurs à plusieurs variables ,

Cela s'écrit : a,b,c= valeur de a, valeur de b, valeur de c

- On peut aussi **échanger** deux variables a,b=b,a

Ex 8 : **Présentation**

Faire un programme de deux lignes maximum, demandant à l'utilisateur ses prénom, nom et âge puis affichant un message de type« Bonjour *prénom nom*, vous avez *âge* ans »

Ex 9 : Deviner ce qu'affichent ces programmes, puis vérifier.

a,b=3,2 a,b=a+b,a-b print(a,b) #…… ?	a,b=3,2 a=a+b b=a-b print(a,b) #…….. ?

Ex 10 : Aire d'un rectangle

Écrire un programme qui demande « Que vaut la largeur ? » puis « Que vaut la longueur ? » et affiche comme réponse « L'aire du rectangle vaut » suivi du résultat.

Prolongements favoris conseillés

- Exercices en ligne (s'inscrire et suivre au fur et à mesure du cours) : Cercles informatiques FranceIOI

- Manuels : EduPython (V. Maille A.Baraquin) , Apprendre à programmer avec Python (G. Swinnen)

- **Sites** : MaTex (J.Cottin) , fsincère , OpenClassRooms